Visual Servo Control with Artificial Neural Network

C.C. Chan
Department of Electrical & Electronic Engineering
University of Hong Kong
Pokfulam Road
HONG KONG

E.W.C. Lo
Division of Technology
City Polytechnic of Hong Kong
83 Tat Chee Avenue, Kowloon
HONG KONG

Abstract — In this paper a robotic manipulator system is presented, which uses visual feedback to control both the position and orientation of the manipulator. In this system, visual data from the camera are directly used in the servo control process of the manipulator, without the calculations of the inverse kinematics. This approach is achieved by the use of an artificial neural network (ANN) to learn the relationship between the position & the orientation of the manipulator and the joint angles. As there is no pre-stored parameters of the manipulator, the system is adaptable to slight changes in the kinematics of the manipulator or the working environment. Hence, this technique should able to extend the application of robotic technology in some special industrial areas. The results of computer simulations of this system is also presented in the paper.

I. INTRODUCTION

In recent years, systems which integrate both visual sensors and robots together have received a lot of attention, especially in the field of intelligent robots [1] [2] [3]. This type of systems can cope with many problems which limit applications of current robots. In this paper, a control scheme for robotic manipulator system will be presented, which makes use of visual information to position and orientate the end-effector with respect to a target object. And an artificial neural network (ANN) is used as a core part of the control.

In the control scheme, a camera is mounted on the end-effector of the manipulator. The camera captures the image of the target object, as seen from the end-effector, when the end-effector moves. This image is termed as 'current image'. The position and the orientation of the target object with respect to the base of the robot are assumed to be unknown in advance, but the given desired position and orientation of the end-effector with respect to the target object are 'measured'. The 'measurements' are in terms of a visual image of the target object as seen from the end-effector, when end-effector is located at the given desired position and orientation with respect to the target object. This visual image is termed as 'target image'.

The control scheme then derives manipulator command signals by comparisons between the 'target image' and the 'current image'. The scheme directly integrates visual data into the servoing process without subdividing the process into determination of object position and orientation, and inverse kinematic calculations. This scheme is made possible by using an artificial neural network (ANN) to learn the relationships between the suitable manipulator commands and results of the visual comparisons between the target image and the current image.

The learning process of the ANN is arranged to be in unsupervised mode. In the learning phase, small random movement commands are sent by the controller and executed by the manipulator. The changes between the current images before and after each small manipulator movement are used as inputs to train the ANN, while the corresponding movement commands are the corresponding desired outputs of the ANN. After sufficient self-training, the system may enter into a execution phase with the differences between the current image and target image as the inputs of the ANN. Then the ANN gives the desired manipulator movement commands. Self-learning could be continued during the execution phase and/or re-initiated after a period of execution phase.

This system is suitable for industrial applications where repetitive movements of manipulator are required, but there are slight differences in each time. Such as the starting positions and/or orientations of the end-effector with respective to the final position / orientation are different in each time. Or the kinematics of the manipulator changes slightly in each time due to the loading effect or a flexible member in the structure of the manipulator. For these kinds of applications, the learning time and learning efforts required will decrease, as the manipulator executes more and more.

II. SCHEME OF CONTROL

The simulated system is assumed to work in the following situation. There are many objects (all are basically identical) come into the working area of the manipulator, one at a time. When an object comes in, its orientation and position relative to the robot is completely arbitrary. The manipulator is then required to move the end-effector to a designated position and orientation (the target position and orientation) relative to the object, before any other operations can be done (for instance, drill a hole on a specific location of the object or grip the object). Fig. 1 illustrates this situation.

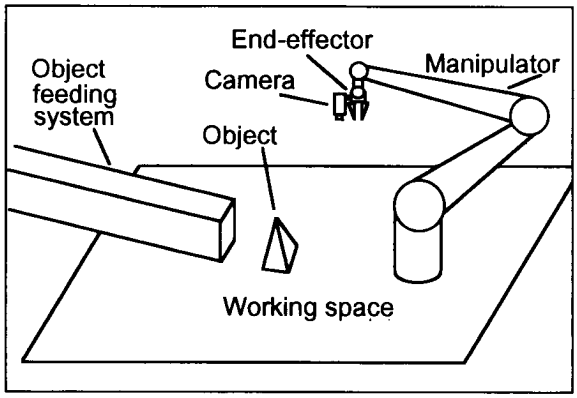

Fig. 1 : The situation under consideration

The camera mounted on the end-effector takes a visual image (the 'current images') of the target object when the robotic manipulator moves. There are visual cues on each of the target object which are used to identify some specified points on the target object. In order to solve the ambiguous problem of position and orientation, the number of visual cues on the target object should be at least four, and they should not be coplanar, geometrically [4]. Different colours may be used for each of the visual cues, in order to distinguish them from one and the other. If there are four visual cues, the natural choice is a set of the three primary colours (red, yellow and blue) plus white, each colour is for each of the four cues. Any simple algorithm (for instant, simple convolution and threholding operations) can be used to extract the coordinate pairs of these visual cues on the visual image from video signals of the camera. Such video preprocessing will not be addressed in this paper.

An example of a 'current image' is shown in Fig. 2. In this example, it is assumed that there are four visual cues on the object. The points of the visual cues on the 'current image' is denoted by, p_n, and with coordinate pairs of (x_n, y_n), n = 1, 2, 3, or 4.

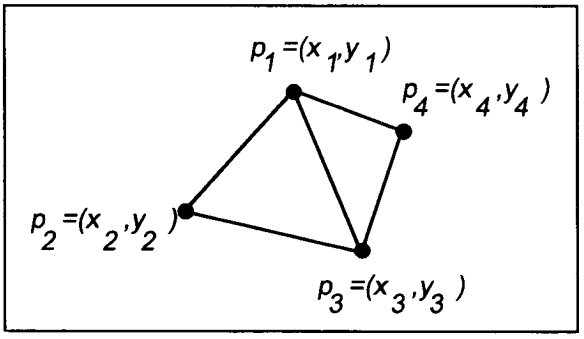

Fig. 2. An example of current image of an object for the camera

The required target position and orientation of the end-effector with respect to the object is expressed to the system in form of the coordinate pairs of the visual cues on the visual image when the end-effector is at the target position. Hence, one of the ways to input the information of the target position into the system is simply move the end-effector manually to the target position and let the camera to take an image of object (called the 'target image'). An example of such a 'target image' of that of Fig. 2 is shown in Fig. 3. The points of the visual cues on the 'target image' is denoted by, P_n, and with coordinate pairs of (X_n, Y_n), n = 1, 2, 3, or 4.

The ANN is used here to generate a set of suitable commands to the manipulator, in form of changes in the joint angles ($\Delta\theta_1$ to $\Delta\theta_6$), to move the end-effector to the 'target position' (it is assumed the manipulator has six degrees of freedom). Hence the block diagram of the system can be expressed as that shown in Fig. 4.

In this paper, we will present a simulation study of such a system for a robotic manipulator with only three joints, hence the outputs of the ANN are the changes required in the three joint angles, $\Delta\theta_1$, $\Delta\theta_2$ and $\Delta\theta_3$.

III. ARTIFICIAL NEURAL NETWORK

The ANN is a three-layer back-propagation network. The number of nodes in the hidden layer is 12. The number of inputs to the ANN is eight, they are the differences of the coordinate values between p_1 to p_4 and P_1 to P_4 (Δx_i, Δy_i, with i = 1, 2, 3 and 4). That is,

$$\Delta x_i = X_i - x_i, \quad \text{and} \quad \Delta y_i = Y_i - y_i, \tag{1}$$

with i = 1, 2, 3 and 4.

While the number of output nodes is three, and the outputs are the commands to the joint motors, $\Delta\theta_1$, $\Delta\theta_2$ and $\Delta\theta_3$. Fig. 5 shows the schematic of the ANN used.

The ANN is simulated by a software program written in Borland C++ version 4 running on a P-5 ,100 MHz, based computer.

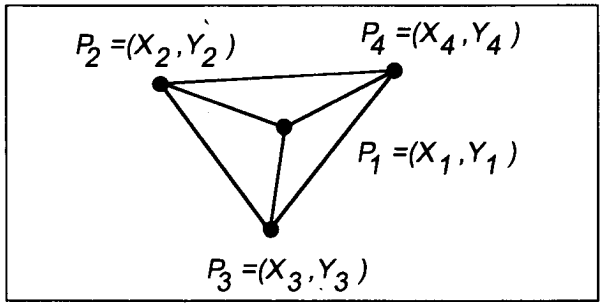

Fig. 3. An example of target image of an object for the camera

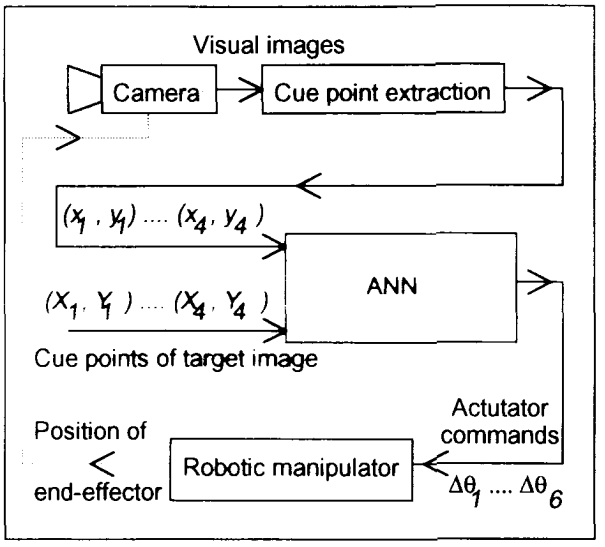

Fig. 4 The block diagram of the system

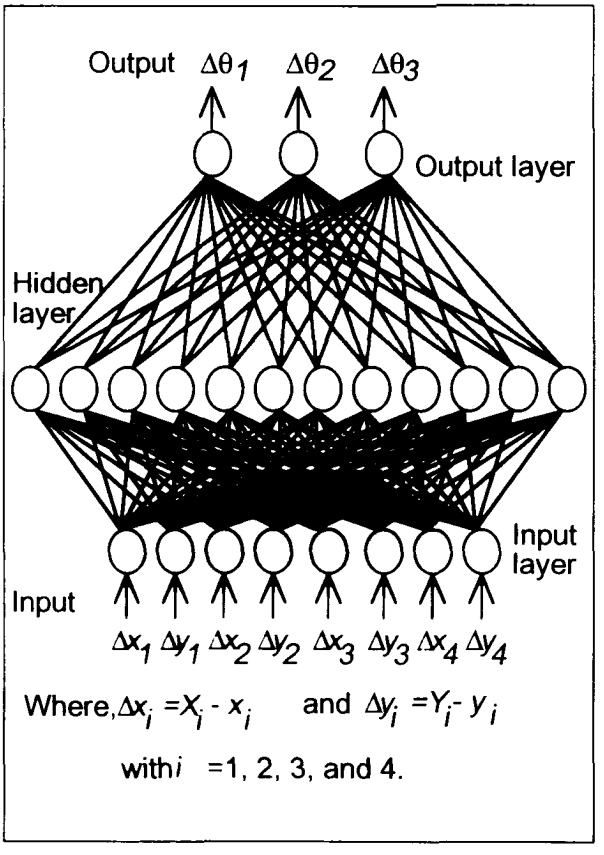

Fig. 5 The artificial neural network used

The Learning Phase

The learning of the system occurs by a sequence of trial movements without the need of an external supervisor. The ANN learns the relationship between the changes in joint angles and the changes in the coordinate values of the visual cue points by sending commands of small random joint angle movement ($\Delta\theta^*_1$, $\Delta\theta^*_2$ and $\Delta\theta^*_3$) to the manipulator. Then the system observes the changes of the location of the cue points (Δx_i and Δy_i) on the visual images. These values of Δx_i and Δy_i are subsequently fed into the ANN and the ANN adjusts its weights to minimize the errors, ε, which are the differences between its outputs and the actual changes in the joints angles. That is,

$$\varepsilon_i = (\Delta\theta_i - \Delta\theta^*_i), \qquad (2)$$

$$i = 1, 2, 3 \text{ and } 4$$

The processes of this learning phase are represented diagramatically in Fig. 6.

More details on the learning algorithm will be presented in the next section

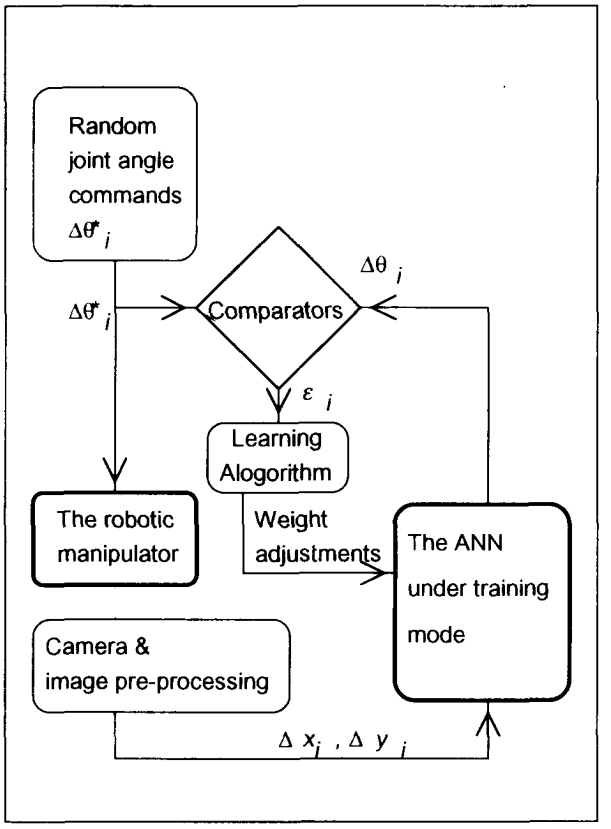

Fig. 6. The learning phase of the ANN

The Execution Phase

During the execution phase, coordinate pairs of the cue points *p1* to *p4* on the 'current image' and those in the 'target image' (*P1* to *P4*) are compared and their differences (Δx_i, Δy_i) are fed into the ANN. The outputs of the ANN are then supposed to be the required joint angle movements which will bring the end-effector to the 'target position'. These joint angle commands are then fed into the motor controller of the robotic manipulator. At the end of the command, the system will check the new values of (Δx_i, Δy_i) again. The above steps are repeated until the 'total error', E, are within an acceptable range. The 'total error' is defined as the root-mean-square value of the eight values of Δx_i and Δy_i. That is,

$$E = \sqrt{\frac{\sum_{i=1,2,3,4} \Delta x_i^2 + \Delta y_i^2}{8}}. \qquad (3)$$

The processes of this execution phase are represented diagramatically in Fig. 7.

During the execution phase, the self learning can still be carried out in parallel, as the ANN can use the observed values of (Δx_i, Δy_i) and its accumulated output values of $\Delta\theta_i$, as the training set. The system may be required to re-enter into the learning phase (obviously, at another initial position) at the middle of an execution phase, if it finds itself is at a total loss.

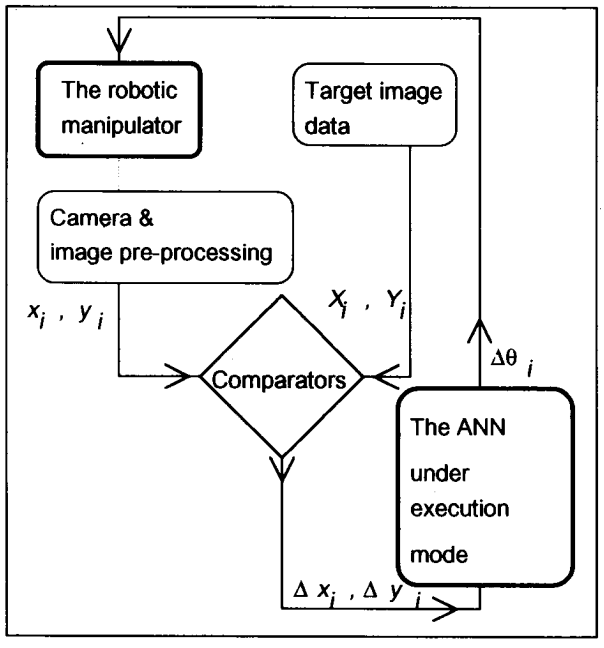

Fig. 7. The execution phase of the ANN

IV. LEARNING ALGORITHM

The back-propagation with steepest-descent learning procedure [5] [6] is used in the training phase of the ANN. At each step of learning, the weights are adjusted in the direction (within the weight space) in which the output errors reduces most rapidly. That is

$$\Delta w = -\rho \nabla J(w), \qquad (4)$$

where, w is the vector of weights,
Δw is the change in the weight vector,
ρ is the learning-rate parameter,
$J(w)$ is the error function to be minimized, and
∇ is the gradient operator.

The gradient $\nabla J(w)$ is the vector of the first partial derivatives of $J(w)$ in the weight space:

$$\nabla J(w) = \left(\frac{\partial J(w)}{\partial w_1}, \ldots, \frac{\partial J(w)}{\partial w_n}\right). \qquad (5)$$

Momentum terms are added in the algorithm to prevent local minima of the $J(w)$ function in the weight space.

The $J(w)$ function in a step of learning is the mean square error of all the root-mean-square values of ε_i. The initial values of the weights are randomly generated. In each step of learning, the weights are adjusted after all the patterns of the training set are presented to the system.

V. RESULTS OF SIMULATIONS

Simulation results for a robotic manipulator with three degrees of freedom are presented here. The target object is a tetrahedron, with a cue point at each of the four corners. A training set of 50 random small joint angle movements (less than 10 degrees in each of the joint angles) are used to train the ANN. The typical learning curve resulted is shown in Fig. 8. It can be seen that the average error is less than 0.05 after 10,000 steps of learning, and it is further reduced to about 0.01 after 30,000 steps of learning. Hence the amount of learning time required depends of the accuracy set.

In the simulation of the execution phase, very satisfactory results are achieved when the target position are in the neighborhood of the initial position (within 10 degrees of movement in each joint angle). The target position can be reached in one single execution.

Fig. 9 shows the typical results in a execution phase, in which relatively large joint angle movements are involved. The target image is not closely achieved after the first set of joint angle movements are executed. However, after the fourth set of movements, the target position is very closely reached, with 'total error' (E) less than 0.01 (assume the 'camera screen' is of the dimension 2 x 2).

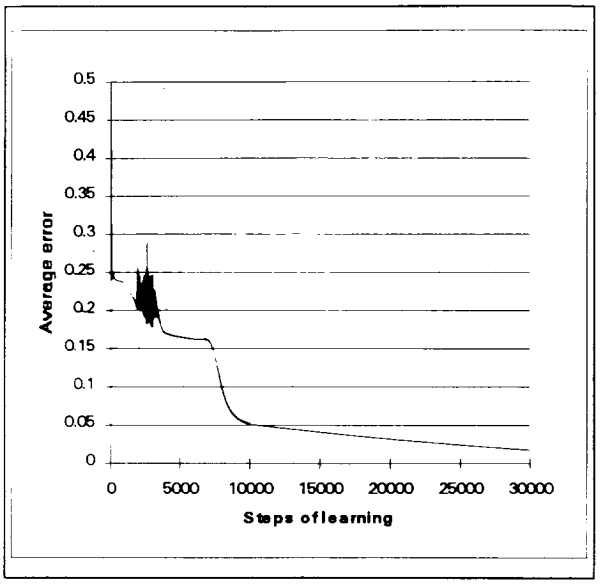

Fig. 8. The learning curve of the ANN

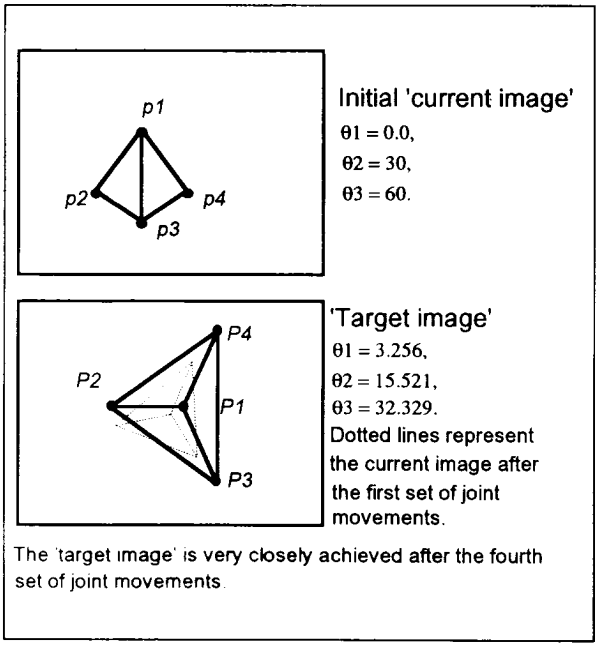

Fig. 9. Results of the computer simulations of the system

The reason of why the target position cannot be achieved in first execution is that, the joint angles in the training set are small (less than 10 degrees) in order to lower the number of patterns in the set, hence reduce the training time. Therefore, when in the cases where large angle movements are required, the outputs of the ANN are not as accurate as in the training situation. Hence this is a compromise between the scope of joint angle coverage and the training time. In most of the applications, it is the authors' view that this compromise is acceptable. Hence the simulations are considered as successful.

VI. CONCLUSIONS

In this paper, a control scheme of robotic manipulators based on visual feedback and artificial neural network is presented. In the scheme, the complex and non-linear relationships between the visual data and the joint angle control commands are learned by the ANN. This approach eliminates the complex kinematic and geometric calculations by transforming it into a mapping learnt in the ANN. The system is also adaptive to slight changes in the working environment or in the structure of the robotic manipulator. The feasibility and validity of the scheme are verified by computer simulations.

Implementations of the system into industrial hardware and simulations of more complex situations are now under investigation.

VII. REFERENCES

[1] D. Vernon, "A system for robot manipulation of electrical wires using vision", *Robotica*, 1990, Volume 8, pp 47-60

[2] D. C. Tseng, Z. Chen, "Computing location and orientation of polyhedral surfaces using a laser-based vision system", *IEEE Transactions on Robotics And Automation*, Volume 7, No. 6, December 1991, pp. 842-846.

[3] T.M. Martinetz, H.J. Ritter, K.J. Schulten, "Three-dimensional neural net for learning visuomotor coordination of a robot arm", *IEEE Transactions on Neural Network*, Volume 1, No. 1, March 1990, pp. 131-140.

[4] Y. Hung, P. Yeh, D. Harwood, "Passive ranging to known planar point sets," in *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, pp. 80-84.

[5] B. Widrow and M.E. Hoff, "Adaptive switching circuits" in *Convention Record of the WESCON* , Part IV, 1960, pp. 96-104.

[6] D.E. Rumelhart, G.E. Hinton & R.J. Williams, "Learning internal representations by error propagation", in *Technical Report 8506 of the Institute for Cognitive Science*, UCSD.